

- [67] M. Musch, C. Wressnegger, M. Johns, and K. Rieck, “New Kid on the Web: A Study on the Prevalence of WebAssembly in the Wild,” in *Detection of Intrusions and Malware, and Vulnerability Assessment - 16th International Conference, DIMVA*, vol. 11543, pp. 23–42, 2019.
- [68] S. Bhansali, A. Aris, A. Acar, H. Oz, and A. S. Uluagac, “A First Look at Code Obfuscation for WebAssembly,” in *WiSec ’22: 15th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, pp. 140–145, 2022.
- [69] B. Baudry and M. Monperrus, “The Multiple Facets of Software Diversity: Recent Developments in Year 2000 and Beyond,” *ACM Comput. Surv.*, vol. 48, no. 1, pp. 16:1–16:26, 2015.
- [70] K. Pohl, G. Böckle, and F. van der Linden, *Software Product Line Engineering - Foundations, Principles, and Techniques*. Springer, 2005.
- [71] S. Sidiroglou-Douskos, S. Misailovic, H. Hoffmann, and M. C. Rinard, “Managing Performance vs. Accuracy Trade-offs With Loop Perforation,” in *SIGSOFT/FSE’11 19th ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE-19) and ESEC’11: 13th European Software Engineering Conference (ESEC-13)*, pp. 124–134, 2011.
- [72] Avizienis and Kelly, “Fault Tolerance by Design Diversity: Concepts and Experiments,” *Computer*, vol. 17, no. 8, pp. 67–80, 1984.
- [73] T. Y. Chen, F. Kuo, R. G. Merkel, and T. H. Tse, “Adaptive Random Testing: The ART of test case diversity,” *J. Syst. Softw.*, vol. 83, no. 1, pp. 60–66, 2010.
- [74] T. Jackson, *On the Design, Implications, and Effects of Implementing Software Diversity for Security*. PhD thesis, University of California, Irvine, 2012.
- [75] G. R. Lundquist, V. Mohan, and K. W. Hamlen, “Searching for Software Diversity: Attaining Artificial Diversity through Program Synthesis,” in *Proceedings of the 2016 New Security Paradigms Workshop, NSPW ’16*, p. 80–91, 2016.
- [76] P. Koopman and J. DeVale, “Comparing the robustness of POSIX operating systems,” in *Digest of Papers. Twenty-Ninth Annual International Symposium on Fault-Tolerant Computing (Cat. No.99CB36352)*, pp. 30–37, 1999.
- [77] I. Gashi, P. Popov, and L. Strigini, “Fault Diversity among Off-The-Shelf SQL Database Servers,” in *Proceedings of the 2004 International Conference on Dependable Systems and Networks, DSN ’04*, p. 389, 2004.

- [78] J. C. Knight and N. G. Leveson, “An experimental evaluation of the assumption of independence in multiversion programming,” *IEEE Transactions on Software Engineering*, vol. SE-12, no. 1, pp. 96–109, 1986.
- [79] B. Randell, “System Structure for Software Fault Tolerance,” *SIGPLAN Not.*, vol. 10, p. 437–449, apr 1975.
- [80] N. Harrand, *Software Diversity for Third-Party Dependencies*. PhD thesis, Royal Institute of Technology, Stockholm, Sweden, 2022.
- [81] J. V. Cleemput, B. Coppens, and B. D. Sutter, “Compiler Mitigations for Time Attacks on Modern X86 Processors,” *ACM Trans. Archit. Code Optim.*, vol. 8, no. 4, pp. 23:1–23:20, 2012.
- [82] A. Homescu, S. Neisius, P. Larsen, S. Brunthaler, and M. Franz, “Profile-guided Automated Software Diversity,” in *Proceedings of the 2013 IEEE/ACM International Symposium on Code Generation and Optimization, CGO 2013, Shenzhen, China, February 23-27, 2013*, pp. 23:1–23:11, IEEE Computer Society, 2013.
- [83] S. Bhatkar, D. C. DuVarney, and R. Sekar, “Address Obfuscation: An Efficient Approach to Combat a Broad Range of Memory Error Exploits,” in *Proceedings of the USENIX Security Symposium*, 2003.
- [84] S. Bhatkar and D. C. DuVarney, “Efficient Techniques for Comprehensive Protection from Memory Error Exploits,” in *Proceedings of the 14th USENIX*, 2005.
- [85] K. Pettis and R. C. Hansen, “Profile Guided Code Positioning,” in *Proceedings of the ACM SIGPLAN’90 Conference on Programming Language Design and Implementation (PLDI)*, pp. 16–27, 1990.
- [86] S. Crane, A. Homescu, S. Brunthaler, P. Larsen, and M. Franz, “Thwarting Cache Side-channel Attacks Through Dynamic Software Diversity,” in *22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2015*, The Internet Society, 2015.
- [87] A. Romano, D. Lehmann, M. Pradel, and W. Wang, “Wobfuscator: Obfuscating JavaScript Malware via Opportunistic Translation to WebAssembly,” in *2022 IEEE Symposium on Security and Privacy (SP) (SP)*, pp. 1101–1116, may 2022.
- [88] M. T. Aga and T. M. Austin, “Smokestack: Thwarting DOP Attacks with Runtime Stack Layout Randomization,” in *IEEE/ACM International Symposium on Code Generation and Optimization, CGO*, pp. 26–36, 2019.